


Using the Split Method in PowerShell

Rate this article ★★★★★

The Scripting Guys (<https://social.technet.microsoft.com/profile/The+Scripting+Guys>) July 17, 2014
 9 (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comments>)

Summary: Microsoft Scripting Guy, Ed Wilson, talks about using the Split method in Windows PowerShell.

Share 1

0

0

Microsoft Scripting Guy, Ed Wilson, is here. Dude, dude, dude, (or dudette, as the case may be) I still cannot find my teapot after our move. At least I found my kettle to heat water so I can make tea, but unfortunately, without a teapot, I am stuck drinking “bagged” tea leaves. It is not a major crisis yet, but I do miss blending my own teas with my own herbs. And of course, my various tins of teas and herbs are sitting here, just waiting for me to locate the teapot. On the bright side, I did pick up some nice tea bags when the Scripting Wife and I were in Europe, so it is not a total loss. In fact, I got a rather nice oolong tea in little triangle sachets that is actually not bad.

Dude, let's split

The **Split** method from the **System.String** class is not a static method. This means that when I have a string, I can gain access to the **Split** method. But if I do not have a string, I cannot access it. The MSDN documentation for the **String.Split Method** ([http://msdn.microsoft.com/en-us/library/System.String.Split\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/System.String.Split(v=vs.110).aspx)) lists six overloads for this method. So I have a lot of flexibility on how I might want to use the method. I will not discuss all six overloads today, but I will discuss the three main ways of using the method:

- Split on an array of Unicode characters
- Split on an array of strings with options
- Specify the number of elements to return

The additional ways of calling the method will behave in a similar fashion.

Split on an array of Unicode characters

One of the cool things about the **Split** method is that it will accept an array of things upon which to split. Therefore, I can split on one or more Unicode characters.

Note A handy list of Unicode characters and their associated code values

(http://en.wikipedia.org/wiki/List_of_Unicode_characters#Basic_Latin) appears on Wikipedia. When using them in Windows PowerShell, remove the U+ and change it to 0X, then call it with **[Char]**. For example, the right curly brace is **[char]0X007D**.

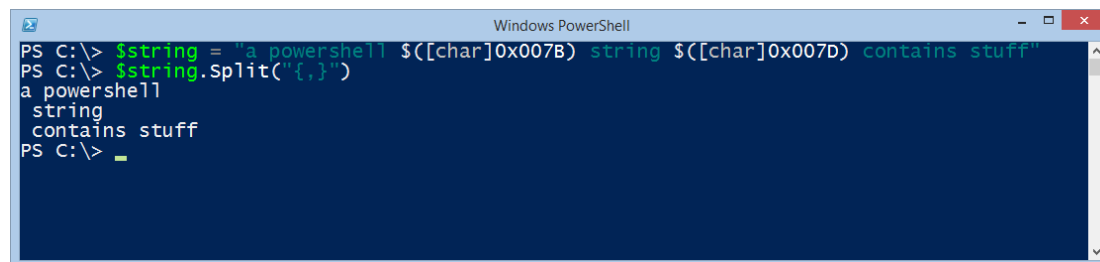
The first thing I need is a string with Unicode characters embedded inside it. So here is my string:

```
$string = "a powershell $([char]0x007B) string $([char]0x007D) contains stuff"
```

Now I need to create an array of two characters to use to split the string. Here is what I come up with the first time:

```
$string.Split("{,}")
```

And this command works. Here is a screenshot of the string, my split characters, and the associated output:



```
Windows PowerShell
PS C:\> $string = "a powershell $([char]0x007B) string $([char]0x007D) contains stuff"
PS C:\> $string.Split("{,}")
a powershell
string
contains stuff
PS C:\>
```

(<https://msdnshared.blob.core.windows.net/media/TNBlogFS/prod.evlo.blogs.technet.com/CommunityServer.Blogs.Components.Web/7-17-14-01.png>)

What is cool is that I can do this same thing in reverse. Here is an example of reversing the process:

```
PS C:\> $string2 = "a powershell {string} contains stuff"
```

```
PS C:\> $string2.Split([char]0x007B, [char]0x007D)
```

```
a powershell
```

```
string
```

contains stuff

```
PS C:\>
```

Split on an array of strings with options

The option to specify an array of strings to use for splitting a string offers a lot of possibilities. The **StringSplitOptions** enumeration also offers a way to control the return of empty elements. The first thing I need to do is to create a string. Here is my string:

```
$string = "This string is cool. Very cool. It also rocks. Very cool. I mean dude. Very cool. Very cool."
```

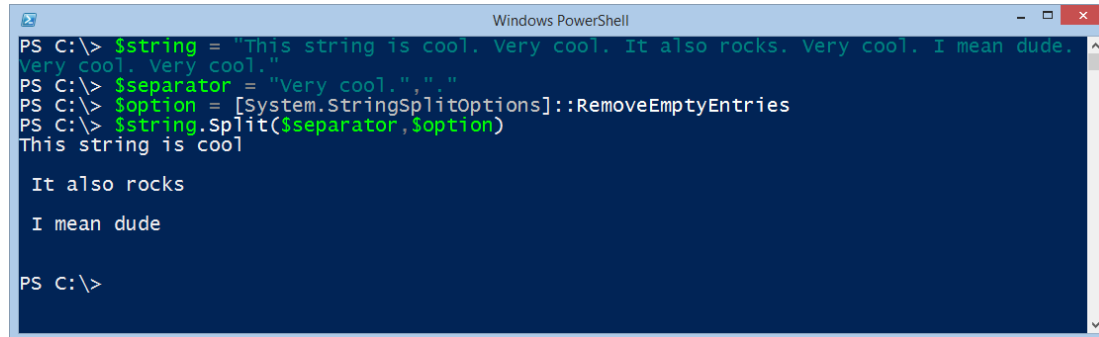
Now, I need to specify a separator. To do this, I will assign an array of strings to the **\$separator** variable as shown here:

```
$separator = "Very cool.", "."
```

Now, I need to create my **StringSplitOptions**. There are two options: **None** and **RemoveEmptyEntries**. The first time I run the command, I will remove the empty entries. So here is that command:

```
$option = [System.StringSplitOptions]::RemoveEmptyEntries
```

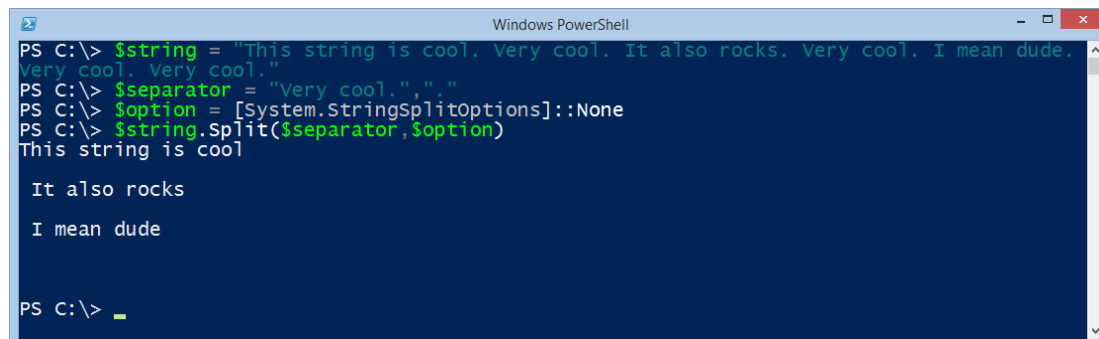
The command and the output from the command appears here:



```
Windows PowerShell
PS C:\> $string = "This string is cool. Very cool. It also rocks. Very cool. I mean dude. Very cool. Very cool."
PS C:\> $separator = "Very cool.", "."
PS C:\> $option = [System.StringSplitOptions]::RemoveEmptyEntries
PS C:\> $string.Split($separator,$option)
This string is cool
It also rocks
I mean dude
PS C:\>
```

(<https://msdnshared.blob.core.windows.net/media/TNBlogFS/prod.evlo.blogs.technet.com/CommunityServer.Blogs.Components.Web/7-17-14-02.png>)

When I change the option to **None**, I see that there is one extra space at the end of the line. This happens because the words "Very Cool." appear twice at the end of the string. This is shown here:



```
Windows PowerShell
PS C:\> $string = "This string is cool. Very cool. It also rocks. Very cool. I mean dude. Very cool. Very cool."
PS C:\> $separator = "Very cool.", "."
PS C:\> $option = [System.StringSplitOptions]::None
PS C:\> $string.Split($separator,$option)
This string is cool
It also rocks
I mean dude
PS C:\>
```

(<https://msdnshared.blob.core.windows.net/media/TNBlogFS/prod.evlo.blogs.technet.com/CommunityServer.Blogs.Components.Web/7-17-14-03.png>)

Specify the number of elements to return

It might be useful to return only a certain number of elements from a string. Here is an example using a string array as a separator:

```
$string = "This string is cool. Very cool. It also rocks. Very cool. I mean dude. Very cool. Very cool."
```

```
$separator = "Very cool.", "."
```

```
$option = [System.StringSplitOptions]::RemoveEmptyEntries
```

```
$string.Split($separator,2, $option)
```

The first time I ran the command, it generated an error message. The reason is that I added the number of elements to return to the end of the method call. The last position is for the **Split** option. Here are the commands and the associated output:

```

PS C:\> $string = "This string is cool. Very cool. It also rocks. Very cool. I mean dude.
Very cool. Very cool."
PS C:\> $separator = "Very cool.", "."
PS C:\> $option = [System.StringSplitOptions]::RemoveEmptyEntries
PS C:\> $string.Split($separator, $option, 2)
Cannot convert argument "options", with value: "2", for "Split" to type
"System.StringSplitOptions": "Cannot convert value "2" to type
"System.StringSplitOptions" due to enumeration values that are not valid. Specify one of
the following enumeration values and try again. The possible enumeration values are
"None, RemoveEmptyEntries".
At line:1 char:1
+ $string.Split($separator, $option, 2)
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [], MethodException
+ FullyQualifiedErrorId : MethodArgumentConversionInvalidCastArgument

PS C:\> $string.Split($separator, 2, $option)
This string is cool
Very cool. It also rocks. Very cool. I mean dude. Very cool. Very cool.
PS C:\>

```

(<https://msdnshared.blob.core.windows.net/media/TNBlogFS/prod.evlo.blogs.technet.com/CommunityServer.Blogs.Components.Web/7-17-14-04.png>)

That is all there is to using the **Split** method. String Week will continue tomorrow when I will talk about more string stuff.

I invite you to follow me on Twitter (<http://bit.ly/scriptingguytwitter>) and Facebook (<http://bit.ly/scriptingguyfacebook>). If you have any questions, send email to me at scripter@microsoft.com (<mailto:scripter@microsoft.com>), or post your questions on the Official Scripting Guys Forum (<http://bit.ly/scriptingforum>). See you tomorrow. Until then, peace.

Ed Wilson, Microsoft Scripting Guy

Tags Scripting Guy! (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/scripting-guy/>) scripting techniques (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/scripting-techniques/>) Strings (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/strings/>) Windows PowerShell (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/windows-powershell/>)

Comments (9)

Name *

Email *

Website

☐ Save my name, email, and website in this browser for the next time I comment.

Post Comment



Anthony Caragol (<https://social.technet.microsoft.com/profile/Anthony+Caragol>)

October 15, 2018 at 10:46 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126813>)

Nice, didn't know about returning a certain number of elements... that's helpful.

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126813#respond>)



hassan sayed issa20014 (<https://social.technet.microsoft.com/profile/hassan+sayed+issa20014>)

October 15, 2018 at 10:46 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126833>)

Hi scripting guy it is a well done job

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126833#respond>)



Anonymous

October 15, 2018 at 10:46 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126853>)

Hi Scripting guy,

I have an string in format

= ABCD : A:Hellofgh.vbs script

I want to get A:Hellofgh.vbs script as output from above.

Please help.

I tried :

```
$a = " = ABCD : A:Helloofgh.vbs script"
```

```
$b = $a.split(':')[1]
```

I get \$b as Helloofgh.vbs script

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126853#respond>)



Vern Anderson (<https://social.technet.microsoft.com/profile/Vern+Anderson>)

October 15, 2018 at 10:46 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126863>)

Thanks Ed!

I wish I would have known about the number between the string and the option before I wrote this blog post

<http://vernanderson.wordpress.com/2014/03/07/split-method-gotchas/> (<http://vernanderson.wordpress.com/2014/03/07/split-method-gotchas/>) Now I know exactly why that happened.

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126863#respond>)



Ed Wilson

July 17, 2014 at 3:53 pm (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126843>)

@Vern Awesome! I am glad that you found the article useful and that you learned something. WOOHOO!!!

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126843#respond>)



Joey

July 18, 2014 at 3:18 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126803>)

Hi, I want to generate user e-mail address from a script. For example user name (Michael), lastname in two phase (John Smith). How can I generate to have

output: michaeljs@xxxx.com (mailto:michaeljs@xxxx.com). That's pick first letter of first word and first letter of second word

from lastname. I have it like this for now: \$e-mail = \$firstname + \$lastname.substring(0,1), output is: michaelj@xxxx.com (mailto:michaelj@xxxx.com). How can

I grab S from John Smith?

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126803#respond>)



alex

October 27, 2014 at 11:07 am (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126823>)

Hi Joey,

I'm sure that there many many ways to do that , and i have just started learning powershell like two weeks ago literally

if you read this topic again in the future , this is one way to solve your dilemma

```
#initial variables
```

```
$firstname = "Michael"
```

```
$lastname = "John Smith"
```

```
[tag:this] will return an array
```

```
$lastname = $lastname -split " "
```

```
[tag:you] can use the array indexes to access "John" and "Smith" separately
```

```
$email = $firstname + $lastname[0].substring(0,1) + $lastname[1].Substring(0,1)
```

```
[tag:Have] fun!
```

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126823#respond>)



DM NZ

April 24, 2015 at 8:56 pm (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-126793>)

Hi ApoorvaW

```
Try $a.substring($a.indexof('.')+1).trim()
```

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=126793#respond>)



Jay_C

October 7, 2015 at 3:26 pm (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/#comment-143361>)

I know how to split a column on delimiters, however, what if I have a column of 6 digit numbers that I need to split in half... for example, split:

123450 into 123 and 450

000065 into 000 and 065

I'm looking for a way to say if the number is 6 digits, split into 2 separate but equal columns.

Thanks!

Reply (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/07/17/using-the-split-method-in-powershell/?replytocom=143361#respond>)

Follow Us



(<https://blogs.technet.microsoft.com/heyscriptingguy/feed/>)

Search MSDN with Bing

☐ Search this blog ☐ Search all blogs



Top Server & Tools Blogs

ScottGu's Blog (<http://weblogs.asp.net/scottgu>)
Brad Anderson's "In the Cloud" Blog (https://blogs.technet.microsoft.com/in_the_cloud/)
Brian Harry's Blog (<http://blogs.msdn.microsoft.com/bharry/>)
Steve "Guggs" Guggenheimer's Blog (<https://blogs.msdn.microsoft.com/stevengu/>)

Subscribe
([http://feeds.feedburner.com/heyscriptingguy](#))

Recent Posts

- PowerTip: Use PowerShell to pick a random name from a list (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/09/15/powertip-use-powershell-to-pick-a-random-name-from-a-list/>) September 15, 2018
- Using PowerShell to create a folder of Demo data (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/09/15/using-powershell-to-create-a-folder-of-demo-data/>) September 15, 2018
- PowerTip: Turn off the power to your computer with PowerShell (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/07/16/powertip-turn-off-the-power-to-your-computer-with-powershell/>) July 16, 2018
- Parse HTML and pass to Cognitive Services Text-to-Speech (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/07/16/parse-html-and-pass-to-cognitive-services-text-to-speech/>) July 16, 2018

Tags

Scripting Guy! (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/scripting-guy/>)
Windows PowerShell (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/windows-powershell/>)
scripting techniques (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/scripting-techniques/>)
PowerTip (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/powertip/>)
guest blogger (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/guest-blogger/>)
VBScript (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/vbscript/>)
getting started (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/getting-started/>)
Weekend Scripter (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/weekend-scripter/>)
Sean Kearney (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/sean-kearney/>)
Office (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/office/>)
Active Directory (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/active-directory/>)
operating system (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/operating-system/>)
storage (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/storage/>) WMI (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/wmi/>)
files (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/files/>) text files (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/text-files/>)
community (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/community/>)
desktop management (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/desktop-management/>)
2011 Scripting Games (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/2011-scripting-games/>)
2012 Scripting Games (<https://blogs.technet.microsoft.com/heyscriptingguy/tag/2012-scripting-games/>)

Related Resources

Script Center Home (<http://technet.microsoft.com/en-us/scriptcenter/default.aspx>)
Scripting Library (<http://technet.microsoft.com/en-us/library/cc498722.aspx>)
Learn to Script (<http://technet.microsoft.com/en-us/scriptcenter/dd793612.aspx>)
Script Repository (<http://www.microsoft.com/technet/scriptcenter/scripts/default.aspx?mfr=true>)
Scripting Forum (<http://social.technet.microsoft.com/Forums/en-US/ITCG/threads>)
2012 Scripting Games (<http://blogs.technet.com/b/heyscriptingguy/archive/2012/02/04/the-2012-windows-powershell-scripting-games-all-links-on-one-page.aspx>)

Archives

September 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/09/>) (2)
July 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/07/>) (2)
June 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/06/>) (4)
May 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/05/>) (5)
March 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/03/>) (3)
February 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/02/>) (4)

December 2017 (<https://blogs.technet.microsoft.com/heyscriptingguy/2017/12/>) (7)
October 2017 (<https://blogs.technet.microsoft.com/heyscriptingguy/2017/10/>) (2)
August 2017 (<https://blogs.technet.microsoft.com/heyscriptingguy/2017/08/>) (2)
July 2017 (<https://blogs.technet.microsoft.com/heyscriptingguy/2017/07/>) (1)
All of 2018 (<https://blogs.technet.microsoft.com/heyscriptingguy/2018/>) (20)
All of 2017 (<https://blogs.technet.microsoft.com/heyscriptingguy/2017/>) (30)
All of 2016 (<https://blogs.technet.microsoft.com/heyscriptingguy/2016/>) (244)
All of 2015 (<https://blogs.technet.microsoft.com/heyscriptingguy/2015/>) (726)
All of 2014 (<https://blogs.technet.microsoft.com/heyscriptingguy/2014/>) (740)
All of 2013 (<https://blogs.technet.microsoft.com/heyscriptingguy/2013/>) (757)
All of 2012 (<https://blogs.technet.microsoft.com/heyscriptingguy/2012/>) (624)
All of 2011 (<https://blogs.technet.microsoft.com/heyscriptingguy/2011/>) (442)
All of 2010 (<https://blogs.technet.microsoft.com/heyscriptingguy/2010/>) (432)
All of 2009 (<https://blogs.technet.microsoft.com/heyscriptingguy/2009/>) (285)
All of 2008 (<https://blogs.technet.microsoft.com/heyscriptingguy/2008/>) (181)
All of 2007 (<https://blogs.technet.microsoft.com/heyscriptingguy/2007/>) (243)
All of 2006 (<https://blogs.technet.microsoft.com/heyscriptingguy/2006/>) (244)
All of 2005 (<https://blogs.technet.microsoft.com/heyscriptingguy/2005/>) (248)
All of 2004 (<https://blogs.technet.microsoft.com/heyscriptingguy/2004/>) (100)

Privacy (<https://privacy.microsoft.com>) Terms of Use (<https://msdn.microsoft.com/cc300389>)
(<https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/EN-US.aspx>)

© 2018 Microsoft